

On Simulated Annealing Dedicated to Maximin Latin Hypercube Designs

Pierre Bergé, Kaourintin Le Guiban,
 Arpad Rimmel, Joanna Tomasik,
 LRI, CentraleSupélec, Université Paris-Saclay
 Bat 650, Rue Noetzlin, 91405 Orsay, France

August 26, 2016

Abstract

The goal of our research was to enhance local search heuristics used to construct Latin Hypercube Designs. First, we introduce the *1D-move* perturbation to improve the space exploration performed by these algorithms. Second, we propose a new evaluation function $\psi_{p,\sigma}$ specifically targeting the Maximin criterion.

Exhaustive series of experiments with Simulated Annealing, which we used as a typically well-behaving local search heuristics, confirm that our goal was reached as the result we obtained surpasses the best scores reported in the literature. Furthermore, the $\psi_{p,\sigma}$ function seems very promising for a wide spectrum of optimization problems through the Maximin criterion.

1 Introduction

The study of complex systems usually requires a considerable computation time. To speed up computations, the system may be replaced by a faster approximating model. To create this model, a set of outcomes for different parameter values is needed. The set of parameter values has an impact on the accuracy of the approximating model. Different sampling methods for this set of parameters has been proposed in [4]. If we note k the number of inputs of the system and n the number of possible values taken by an input variable x_k , the choice of n sample vectors can be represented by points on a hypercube of size n and dimension k . Among the designs proposed, we focus on the Maximin Latin Hypercube Design (LHD).

The LHD implements the Latin constraint: each coordinate in $[1;n]$ must appear only once in every dimension. In other words, the coordinates of any pairs of nodes differ in all dimensions. Moreover, the Maximin constraint means that we search the configuration with the maximal d_{\min} , where d_{\min} is the minimal distance between two points of the design. An instance is defined by

the values of the dimension k and the size n . Consequently, an instance will be noted k/n , for example 10/50.

There are exactly $\binom{n}{2} = \frac{n(n-1)}{2}$ distances between points. They may be ordered $d_1 \leq d_2 \leq \dots \leq d_i \leq d_{i+1} \dots \leq d_{\binom{n}{2}}$, by definition, $d_{\min} = d_1$. In the remainder, we often refer to square values: $D_i = d_i^2$.

As the Maximin LHD problem is believed to be NP-hard, heuristics are widely used to solve it. The use of deterministic methods are, for the moment, rather limited: branch-and-bound was only used with $k \leq 3$ by [8, 9] (a highscore is a minimal maximal distance between any pair of design points for a given instance) . The survey [6] of metaheuristics sums up their performance on Maximin LHD and reports that SA not only outperformed other evolutionary algorithms but also improved many of the previous highscores for $k \leq 10$ and $n \leq 25$. For this reason we choose to work on the improvement of SA applied to Maximin LHDs.

After introducing usual methods used to solve Maximin LHD with SA, we propose a new mutation and a new evaluation function to improve on the best current scores. Eventually, we show how it permits to exceed a great part of the scores presented by the literature. These results should not exclusively be considered in the particular SA context as they may offer the opportunity to boost the performance of local search algorithms for different designs with the Maximin constraint.

N.B. *This reports completes our paper [2] providing the precisions and details which had to be omitted in its camera-ready version due to the page number limit. It should thus be cited together with [2].*

2 Typical approach to solve Maximin LHD with Simulated Annealing

SA is a metaheuristic most commonly used for discrete search spaces inspired by a metallurgical process. It consists in visiting the search space with a perturbation on the configurations and deciding whether the mutated configuration should be selected. That is why it alternates phases of heating and slow cooling to influence this choice: as in physics, by controlling the cooling process, we give an opportunity to the configuration to find the lowest energy.

It is proven that SA converges to the global minimum with the Metropolis probability [1]. Several ingredients are compulsory for SA [3]: a perturbation (a mutation), an evaluation function H_{pot} (also called potential energy) and a temperature decrease $T(k)$, where k is the iteration number. The acceptance probability depends on $T(k)$ and the gap of potential energy between the new and the old configuration. When the current configuration at the iteration number k is ω , a configuration ω' will be accepted with the Metropolis probability:

$$p_k = \min \left(1, e^{\frac{H_{\text{pot}}(\omega') - H_{\text{pot}}(\omega)}{KT(k)}} \right)$$

Constant K is typically fixed to 1. A configuration ω is defined by n points with k coordinates respecting the Latin constraints.

Survey [6] examined several perturbations among which m_2 was the most efficient. It deals with a pair of points: a randomly chosen one and a critical one. A critical point is a point involved in d_{\min} . Mutation m_2 transposes the coordinates of these points in one dimension. The authors of [6] proposed mutation m_3 which is a variant of m_2 as the transposition takes place in the dimension which ensures a better d_{\min} for a subsequent configuration ω' . Mutation m_3 outperformed m_2 for 9/10, 4/25 and 8/20.

Article [6] compared two evaluation functions: $-d_{\min}$ and ϕ_p introduced in [5]:

$$\phi_p = \left(\sum_{i=1}^{\binom{n}{2}} d_i^{-p} \right)^{\frac{1}{p}}. \quad (1)$$

Function ϕ_p is more efficient than $-d_{\min}$ certainly because it takes into account changes on every distance whereas the function $-d_{\min}$ only considers the shortest distance of the configuration. As the paper [6] obtained the best score of the literature, we used the same values to set the values of parameter p .

3 New perturbation targeting LHD

3.1 Principle of the perturbation

We use m_2 as a basis to construct a better performing perturbation. To clarify its principle, we define the notion of the neighborhood:

Definition 1 (Neighbor of a point). *For a given instance k/n , a point p_1 of a configuration ω is a neighbor of the point p_2 if and only if there is a dimension j for which coordinates of these two points are the closest possible. In other terms, $\exists j \in [1; k]$ such that $|p_1(j) - p_2(j)| = 1$.*

The new mutation *1D-move* consists in taking a critical point as before and taking one of its neighbor. Then we exchange the coordinates in one of the dimensions concerned by the neighborhood.

We choose a 3/5 instance to illustrate *1D-move*. Table 1 gives the coordinates of the points of a configuration ω . First, we choose a critical point: d_{\min} is determined by points p_1 and p_2 , so we take p_1 . Points p_2 (on axis x , y and z), p_3 (on axis y) and p_4 (on axis z) are neighbors of p_1 . For the sake of the example, we shall choose p_4 as a neighbor. Then, we exchange the coordinates of p_1 and p_4 on axis z because p_1 and p_4 are neighbors through dimension z . The new configuration is also given in Table 1.

Points	p_1	p_2	p_3	p_4	p_5
x	0	1	2	3	4
y	1	2	0	4	3
z	2	1	4	3	0

Points	p_1	p_2	p_3	p_4	p_5
x	0	1	2	3	4
y	1	2	0	4	3
z	3	1	4	2	0

Table 1: Illustration of *1D-move* with an initial (left) and a following (right) configuration

3.2 Performance Evaluation

1D-move outperforms not only m_2 but m_3 as well. We reproduced the experiments made in [6] keeping the same value of parameter p ($p = 10$) for 4/25, 9/10 and 8/20 to show its performance (Table 2). SA performs a linear thermal descent until temperature $T = 0$ is reached. The initial temperature is set thanks to a series of preliminary runs. We computed 100 effective runs and we present here the average within the 95% confidence interval.

Instance	m_2	m_3	<i>1D-move</i>
4/25	177.59 ± 0.29	177.67 ± 0.29	180.51 ± 0.27
9/10	156.24 ± 0.10	156.06 ± 0.08	156.54 ± 0.06
8/20	431.98 ± 0.61	433.72 ± 0.84	436.20 ± 0.56

Table 2: Performance of SA with different mutations

To explain the efficiency of *1D-move*, we can refer to SA on the Traveling Salesman Problem. Article [7] shows that the perturbations which move the smallest number of edges are the best. *1D-move* modifies the same number of points as m_2 and m_3 . Consequently, $2(n-2)$ distances are modified by all these mutations. However, the changes on distances are smaller with *1D-move* given that modifications on coordinates are ± 1 thanks to the neighborhood property. We prove it with Eq. (2). Our hypothesis is that this specific property explains why *1D-move* is more efficient.

Let us consider a n/k configuration ω and the two mutations m_2 and *1D-move*. We want to prove that the changes due to these two mutations are not necessarily represented by the same order of magnitude. Let us note:

$$m_2 : \omega \longrightarrow \omega' \quad \text{and} \quad \textit{1D-move} : \omega \longrightarrow \omega''.$$

We assume the two mutations translate the point p_2 on a given dimension j . We also take a point p_1 of the configuration which remains invariant with these

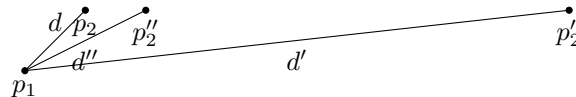


Figure 1: Effect of m_2 and *1D-move*

mutations (p'_1 and p''_1 equal to p_1). The objective is to find a configuration for which $\Delta d = |d_{p'_1, p'_2} - d_{p_1, p_2}|$ is equivalent to n in the m_2 case. This difference is:

$$\Delta d = \left| \sqrt{\sum_{l=1}^k (p'_2(l) - p_1(l))^2} - \sqrt{\sum_{l=1}^k (p_2(l) - p_1(l))^2} \right|.$$

As most of the dimensions are not concerned by this move, we just note:

$$\sum_{l=1, l \neq j}^k (p'_2(l) - p_1(l))^2 = \sum_{l=1, l \neq j}^k (p_2(l) - p_1(l))^2 = a^2.$$

Variable a depends on n and k . If p_1 and p_2 are neighbors regarding all dimensions (except j), $a^2 = k - 1$. We take a configuration for which $a^2 = k - 1$, $p_1(j) = 0$, $p_2(j) = n - 1$ and $p'_2(j) = 1$. This configuration is illustrated in Figure 1.

By separating j from other dimensions, we eventually find:

$$\begin{aligned} \Delta d &= \left| \sqrt{a^2 + (n-1)^2} - \sqrt{a^2 + 1} \right| \\ &= \left| \sqrt{k-1 + (n-1)^2} - \sqrt{k} \right|. \end{aligned} \quad (2)$$

Some configurations respect the property: $k \ll n^2$, for which $\Delta d = O(n)$. This means that the difference between two distances may take values with the order of magnitude n . It is not possible with *1D-move*. Using the triangle inequality and the neighborhood property: $\Delta d \leq 1$.

With m_2 (or m_3 which is more restrictive than m_2), Δd sometimes reaches the order of magnitude n . We showed with $\Delta d \leq 1$ that it was impossible with *1D-move* which allows the local search to be more regular.

4 New evaluation function targeting Maximin

4.1 Presentation of a Maximin effect: narrowing the distribution of distances

We study the properties of distances obtained with the evaluation function ϕ_p in SA solutions. We represent all the distances of a configuration in histograms and identify properties that will allow us to establish a better evaluation function below. From now on, we distinguish three cases relative to values taken by n and k . We note the mean of D for any configuration as $\overline{D}(k, n) = \frac{kn(n+1)}{6}$ as shown in [9].

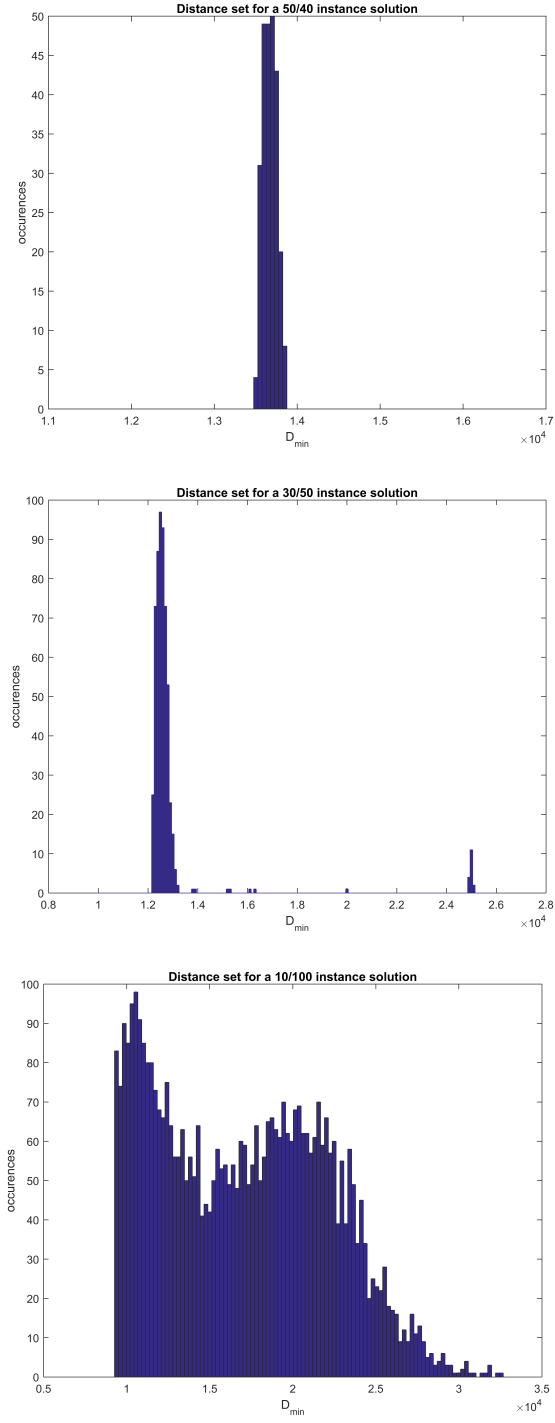


Figure 2: Histograms of distances for 50/40, 30/50 and 10/100 solutions

Case $n \leq k$

In this case (see Figure 2, 50/40), the distances of potential solutions are concentrated around the mean. It is highly probable that two points at random taken will be neighbors. This explains why a point is close to all others in SA solutions and we talk about unimodal distribution. In our example 50/40 in Figure 2, the statistical range of D relative to \bar{D} , $\frac{D_{\max}-D_{\min}}{\bar{D}} = \frac{340}{13667} = 2.5\%$, in fact, is narrow. The rationale for this behavior is that when the number of points is less than the number of dimensions, it happens, in absence of constraints, that all the points are equidistant. Since the Latin constraint has to be respected, the points cannot be exactly equidistant. The distances, however, do not differ significantly.

Case $k \leq n \leq 2k$

In this case (Figure 2, 30/50), distributions are concentrated around two peaks. The first peak is mainly around the average distance (actually, there is a little shift between the peak and the mean because both the peaks preserve \bar{D}) and the second peak is located around the doubled average distance. Much more distances are concerned by the first peak.

We illustrate this phenomenon with the 30/50 instance in Figure 2. We can explain this by the fact that it is possible for this many points to be placed in an hyperoctahedron. In such a geometric object, each point is at the same distance from every other point but one, which is farther away. Thus, the distribution of distances shows two values, with the smaller being represented much more frequently.

In our example, $\bar{D}(30, 50) = 12500$. Concerning the highest peak, the statistical range remains small compared with the mean: the ratio is 7.8%, larger than in the first case for the whole distribution. There are only seven distances located in the interval [13183; 24865]

Case $2k \leq n$

In this last case (Figure 2, 10/100), distances are distributed more uniformly. There is neither a dense peak nor a sparse interval. We observe a decrease of occurrences with an increase in the value of the distance.

Observations and consequences

For the first case, the only peak is naturally thin thanks to SA and particularly ϕ_p action. There is a little point in trying to narrow it more. We note that for the two last cases ($k \leq n$), narrowing differences between distances lead to improve performance. We illustrate this on the 8/20 instance. We represent distance sets of several possible solutions and observe that the best solutions have the most narrowed distributions. We compare two solutions in Figure 3 with $D_{\min} = 421$ and $D_{\min} = 446$ which is the best solution found in [6]. Indeed, we note that $D_{\min} = 446$ has the most narrow peak. We formulate

Inst.	σ	$\phi_{10} \& m_2$	$\psi_{10,\sigma} \& m_2$	$\phi_{10} \& 1D\text{-move}$	$\psi_{10,\sigma} \& 1D\text{-move}$
4/25	70	177.59 ± 0.29	177.98 ± 0.71	180.51 ± 0.27	181.24 ± 0.23
9/10	20	156.24 ± 0.10	156.09 ± 0.06	156.54 ± 0.06	156.49 ± 0.10
8/20	65	431.98 ± 0.61	433.58 ± 0.70	436.20 ± 0.56	445.28 ± 0.45

Table 3: Performance of SA with different setups for evaluation function and mutation

the hypothesis that this property may be beneficial for SA performance. We introduce below a new evaluation function taking into account this aspect.

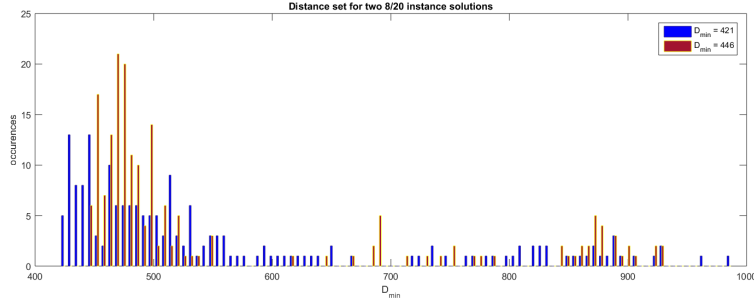


Figure 3: Distance sets of two 8/20 solutions

4.2 Definition of evaluation function ψ

We propose an evaluation function $\psi_{p,\sigma}$ to replace the usual function ϕ_p :

$$\psi_{p,\sigma} = \left(\sum_{i=1}^{\binom{n}{2}} w_i d_i^{-p} \right)^{\frac{1}{p}}, \text{ where } w_i = \frac{1}{\sqrt{\sum_{j=1}^{\binom{n}{2}} e^{-\frac{|D_j - D_i|^2}{\sigma^2}}}}. \quad (3)$$

The idea is to add weights $w_i \geq 1$ for each distance term d_i^{-p} . These weights determine if the distance is close to other ones. If a distance is far from the others, the weight will be high. Consequently, it forces the distances to be close to each other. A single drawback of ψ is its complexity in $\mathcal{O}(n^4)$. There are different ways to reduce this complexity. First, for instance, it is possible to consider only the differences which respect $|D_j - D_i|^2 \leq 5\sigma^2$. In this way, we avoid the calculations of terms that may be considered as negligible ($e^{-5} \ll 1$). Instead of summing up $\binom{n}{2}$ distances, we can randomly choose $\mathcal{O}(n)$ distances D_j .

4.3 Tuning of parameter σ and its justification

Let us focus on the parameter σ : given that we aim at furnishing a large number of scores, we need to tune it in a global way. It must depend directly on n

and k , without preliminary experiments for each instance k/n . Looking at the definition of $\psi_{p,\sigma}$, this variable is introduced in order to regulate the order of magnitude of the exponential term. We see that σ should have approximately the same order of magnitude than the values taken by $|D_j - D_i|^2$.

This is why we try to give the expression of a linear function of k and n which is similar to typical values $|D_j - D_i|^2$. To establish it, we study the variance of a random variable: the tuning of σ is founded on Theorem 2.

Theorem 2. *Let $D(k, n)$ be the random variable representing any square distance in any configuration of instance k/n . We have $D(k, n) \sim \mathcal{N}\left(\frac{kn(n+1)}{6}, g(n)\right)$ with $g(n) \sim \frac{7kn^4}{180} + \mathcal{O}(n^3)$.*

Proof. Thanks to [9], we know that $\mathbb{E}(D(k, n)) = \frac{kn(n+1)}{6}$. We note (P_1, P_2) the random variable that gives any couple of points for n/k . The random variable $D(k, n)$ is a function of (P_1, P_2) . For any $1 \leq j \leq k$, we note $Y(j) = (P_1(j) - P_2(j))^2$ and get $D(k, n) = \sum_{j=1}^k Y(j)$. As $Y(i)$ and $Y(j)$ are independent if $i \neq j$, we note $Y(i) = Y$ to keep the notation simple. If k is high enough, we apply the Central Limit Theorem: $D(k, n) \sim \mathcal{N}\left(\frac{kn(n+1)}{6}, k\text{Var}(Y)\right)$. We focus first on $\mathbb{E}(Y^2) = \mathbb{E}((P_1(j) - P_2(j))^4)$:

$$\mathbb{E}(Y^2) = \frac{\sum_{x=1}^n \sum_{y \neq x} (x - y)^4}{\frac{n(n-1)}{2}} = \frac{2 \left(n \sum_{z=1}^{n-1} z^4 - \sum_{z=1}^{n-1} z^5 \right)}{n(n-1)} = \frac{n^4}{15} + \mathcal{O}(n^3).$$

$$\text{We thus deduce } \text{Var}(Y) = \mathbb{E}(Y^2) - \mathbb{E}(Y)^2 = \frac{n^4}{15} - \frac{n^4}{36} + \mathcal{O}(n^3) \sim \frac{7n^4}{180}. \quad \square$$

We propose a global tuning of σ^2 as a linear function of the variance of our configurations. As computing the variance of a configuration, at every iteration, would be expensive, we formulate the hypothesis that the variance of the square distances set of the SA solutions follows the function $g(n)$ above. The idea of the tuning is to consider σ^2 linearly dependent on the variance of the random variable $D(k, n)$. In the weights w_i , we compare the difference between the current distance and an extra one with σ by calculating $\frac{|D_j - D_i|^2}{\sigma^2}$ in order to identify which differences $D_j - D_i$ have to be taken into account.

In the case $n \geq 2k$, we assume $\sigma^2 = ckn^4$. According to several experiments series, we identify a good compromise with $c = \frac{1}{300}$.

In the case $n \leq k$, leading to unimodal distributions, ψ does not bring more interesting results than ϕ . It is equivalent to assuming c to be very large ($c \rightarrow \infty$).

Finally, the case $k \leq n \leq 2k$ which is an intermediary of the two previous cases, can be tuned with $\sigma = 2ckn^4$. This proposition does not obviously represent the best tuning for all possible instances but gives an efficient and simple solution for the tuning of σ .

It is necessary to mention that the case $k \leq n \leq 2k$ is the case where tuning is essential: to be as efficient as possible, the value of σ has to be carefully

$n \backslash k$	3	4	5	6	7	8	9	10
3	6	7	8	12	13	14	18	19
4	6	12	14	20	21	26	28	33
5	11	15	24	27	32	40	43	50
6	14	22	32	40	47	54	62	68
7	17	28	40	52	62	72	81	91
8	21	42	50	66	80	91	103	116
9	22	42	61	82	95	114	128	144
10	27	50	82	95	113	134	158	175
11	30	55	82	111	133	157	184	211
12	36	63	94	142	158	184	213	243
13	41	70	107	143	184	214	246	279
14	42	78	109	162	220	247	282	318
15	48	89	135	179	228	281	323	363
16	50	94	154	200	254	328	364	412
17	56	102	163	221	<i>277</i>	343	413	462
18	57	114	176	249	306	376	469	515
19	62	123	193	268	336	408	491	576
20	66	138	210	293	372	448	528	645
21	69	149	232	315	401	482	570	674
22	82	154	246	347	433	525	623	721
23	82	165	260	364	468	566	667	<i>773</i>
24	83	173	276	391	506	609	720	837
25	89	183	294	419	541	657	<i>768</i>	897

Table 4: Highscores obtained with “all purpose” tuning

selected. Table 3 shows the impact of $\psi_{p,\sigma}$ on SA performance with mutations m_2 and *1D-move*. We keep the same experimental setup as in Subsection 3.2: SA makes a thermal linear descent, the results presented come out from 100 runs and the average is within the 95% confidence interval.

In Table 4, we update scores for the same instances as in [6]. The results were produced with 10^7 iterations and $p = 5$. Our function $\psi_{p,\sigma}$ is used when $k \leq n$, ϕ_p elsewhere. We note in bold type improved results and in italics results worse than [6]. For $4 \leq k \leq 8$, the use of *1D-move* and $\psi_{p,\sigma}$ allows us to exceed a large number of scores but this improvement is less significant for other values. For $k = 3$, we suppose that the new tools are not able to outperform previous results because the results are already optimal or very good. For $k = \{9, 10\}$, a credible hypothesis is that the value of $\frac{n}{k}$ is so close to 1 that the effect of $\psi_{p,\sigma}$ is weak. Generally, results could be better with a specifically adapted tuning. Here, we established temperature, p and σ by making compromises between all the instances. However, in a real life case, by treating complex systems, we work on a defined instance with k and n fixed. In such circumstances, we naturally advice to customize the tuning of the different parameters by making preliminary experiments on this very instance. We expect that such an approach

would produce results outperforming those in Table 4.

5 Conclusion

In this article, we introduce new techniques to treat the Maximin LHD construction. The first one is the *1D-move* mutation especially dedicated to the LHD structure. It is very efficient for a local search on LHDs because it makes it possible to follow a step-by-step path on the cost surface without jumping over possible minima. The second tool, the evaluation function $\psi_{p,\sigma}$ directly focuses upon Maximin optimization.

As numerous problems, among them Maximin Designs, involve this criterion, we emphasize that this function can be used for many other applications. In the Maximin LHD context, the function $\psi_{p,\sigma}$ tries to find solutions by narrowing a set of possible distances. SA, with *1D-move* and $\psi_{p,\sigma}$, gave results better than those considered to be “the best known” for the majority of cases without any dedicated tuning.

References

- [1] E. Aarts and P. van Laarhoven. Statistical cooling: A general approach to combinatorial optimization problems. *Philips Journal of Research*, 40(4), 1985.
- [2] P. Bergé, K. Le Guiban, A. Rimmel, and J. Tomasik. Search Space Exploration and an Optimization Criterion for Hard Design Problems. In *Proc. of ACM GECCO*, pages 43–44, July 2016.
- [3] S. Kirkpatrick, D. Gelatt Jr., and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
- [4] M. McKay, R. Beckman, and W. Conover. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21, 1979.
- [5] M. D. Morris and T. J. Mitchell. Exploratory designs for computational experiments. *Journal of statistical planning and inference*, 43:381–402, 1995.
- [6] A. Rimmel and F. Teytaud. A Survey of Meta-heuristics Used for Computing Maximin Latin Hypercube. In *Proc. of EvoCOP*, pages 25–36, 2014.
- [7] P. Tian, J. Ma, and D. Zhang. Application of the simulated annealing algorithm to the combinatorial optimisation problem with permutation property: An investigation of generation mechanism. *European Journal of Operational Research*, 118:81–94, 1999.
- [8] E. R. van Dam, B. Husslage, D. den Hertog, and H. Melissen. Maximin latin hypercube designs in two dimensions. *Operations Research*, 55(1):158–169, 2007.

- [9] E. R. van Dam, G. Rennen, and B. Husslage. Bounds for maximin latin hypercube designs. *Operations Research*, 57(3):595–608, 2009.